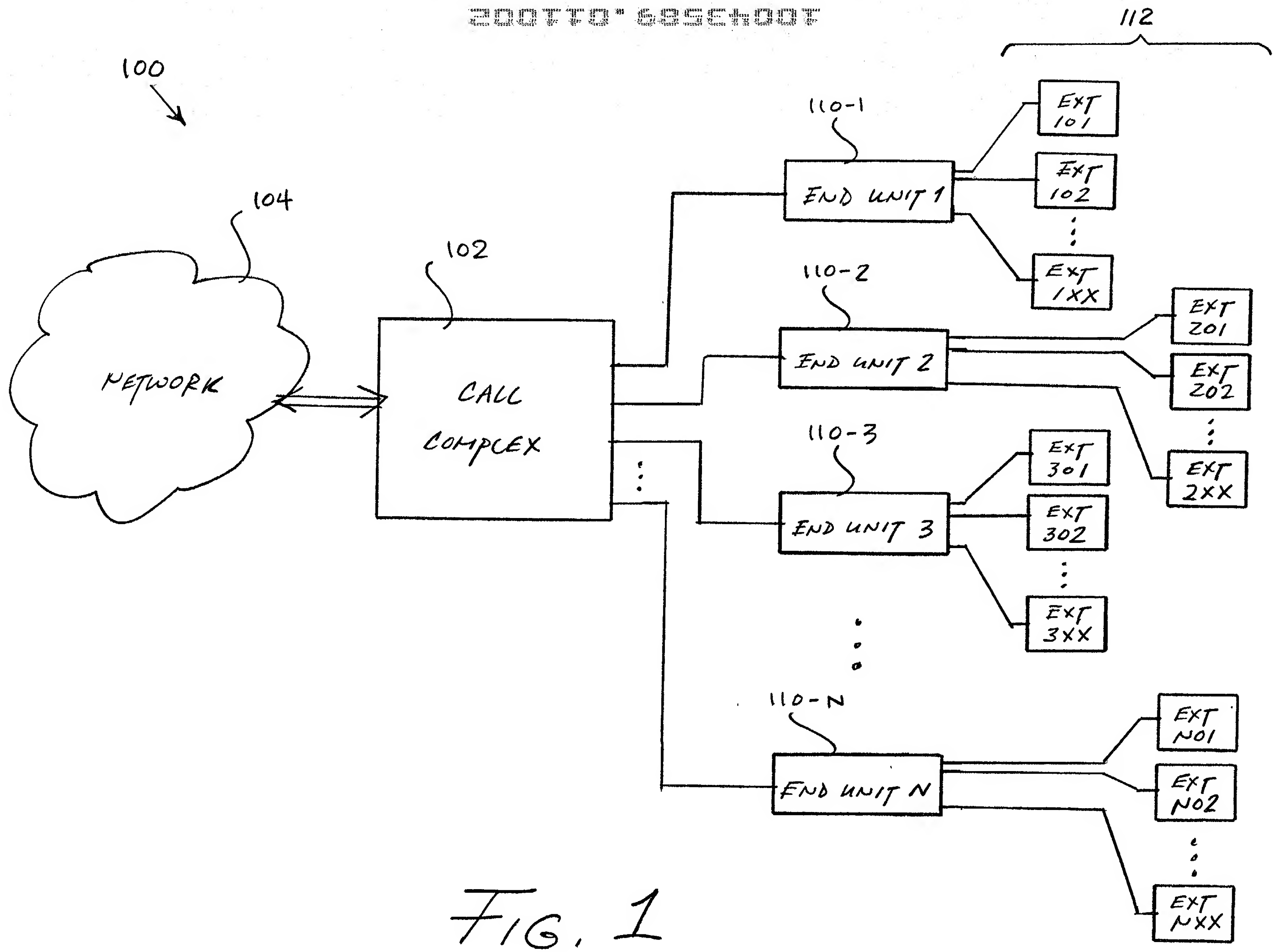
100
104
NETWORK
102
CALL
COMPLEX
110-1
END UNIT 1
110-2
END UNIT 2
110-3
END UNIT 3
110-N
END UNIT N
112
EXT 101
EXT 102
EXT 1XX
EXT 201
EXT 202
EXT 2XX
EXT 301
EXT 302
EXT 3XX
EXT N01
EXT N02
EXT NXX

FIG. 1

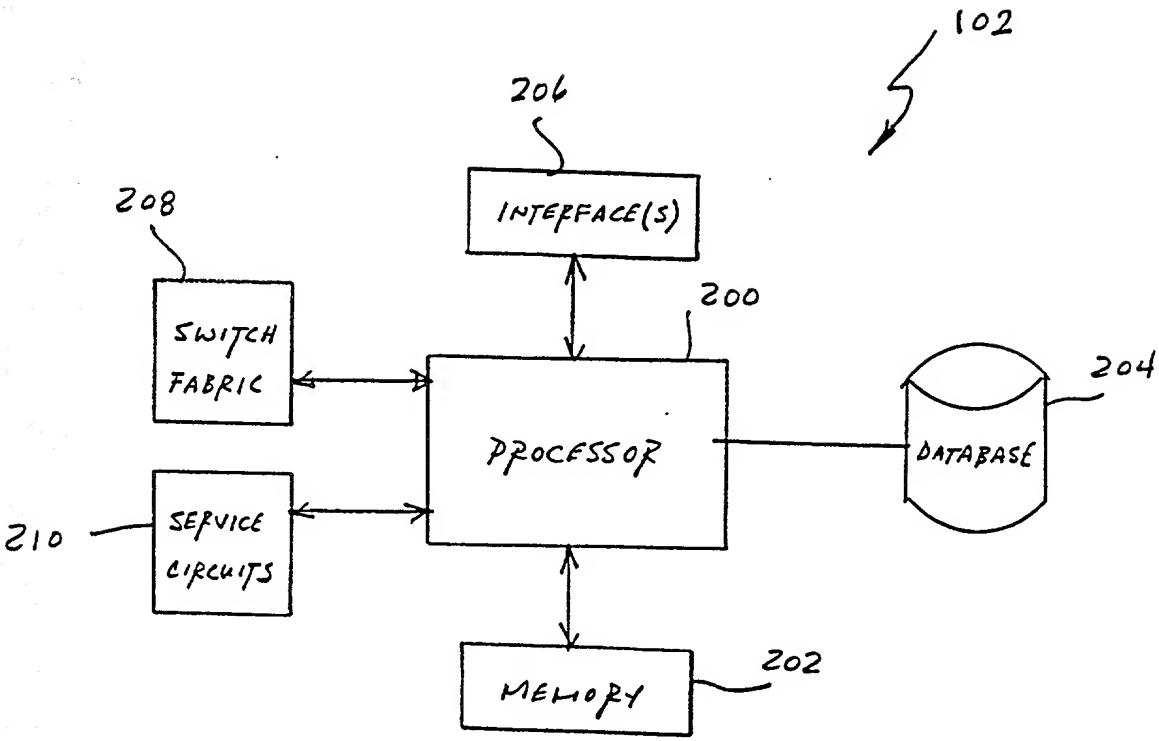
102
206
INTERFACE(S)
208
SWITCH FABRIC
200
PROCESSOR
204
DATABASE
210
SERVICE CIRCUITS
202
MEMORY

FIG. 2

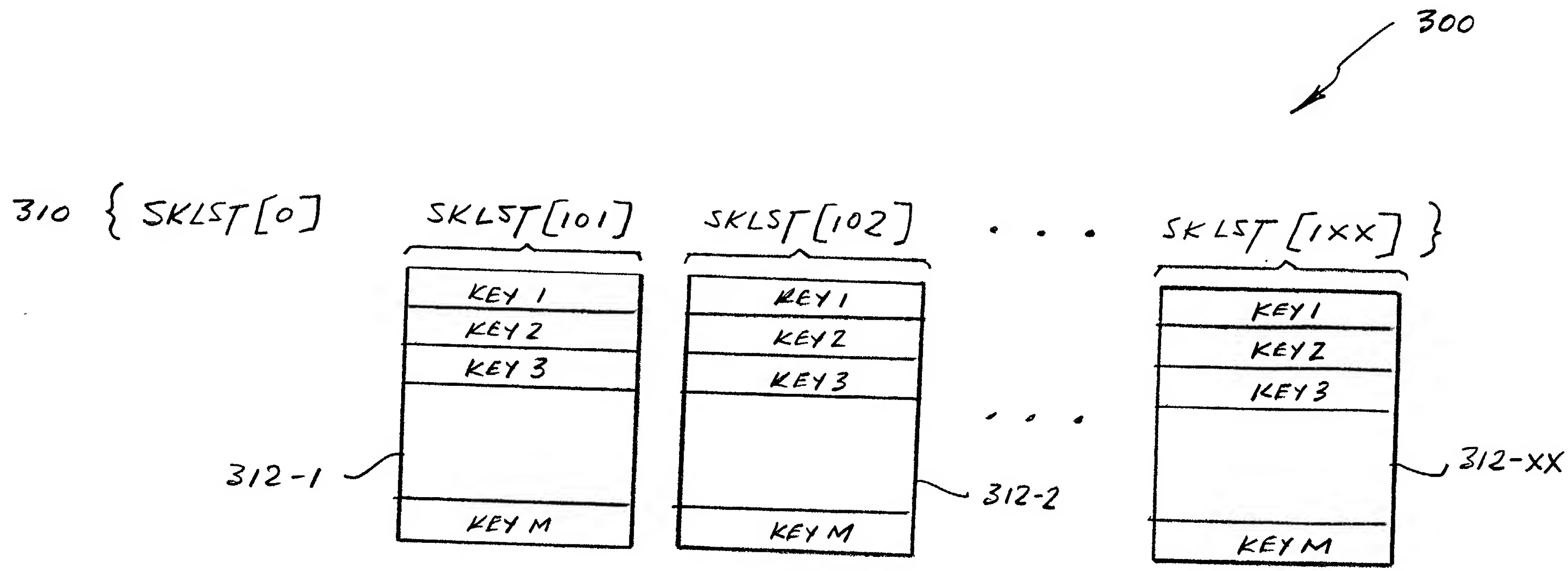
300
310
SKLST[0]
SKLST[101]
SKLST[102]
SKLST[1XX]
KEY 1
KEY 2
KEY 3
KEY M
312-1
312-2
312-XX

FIG. 3

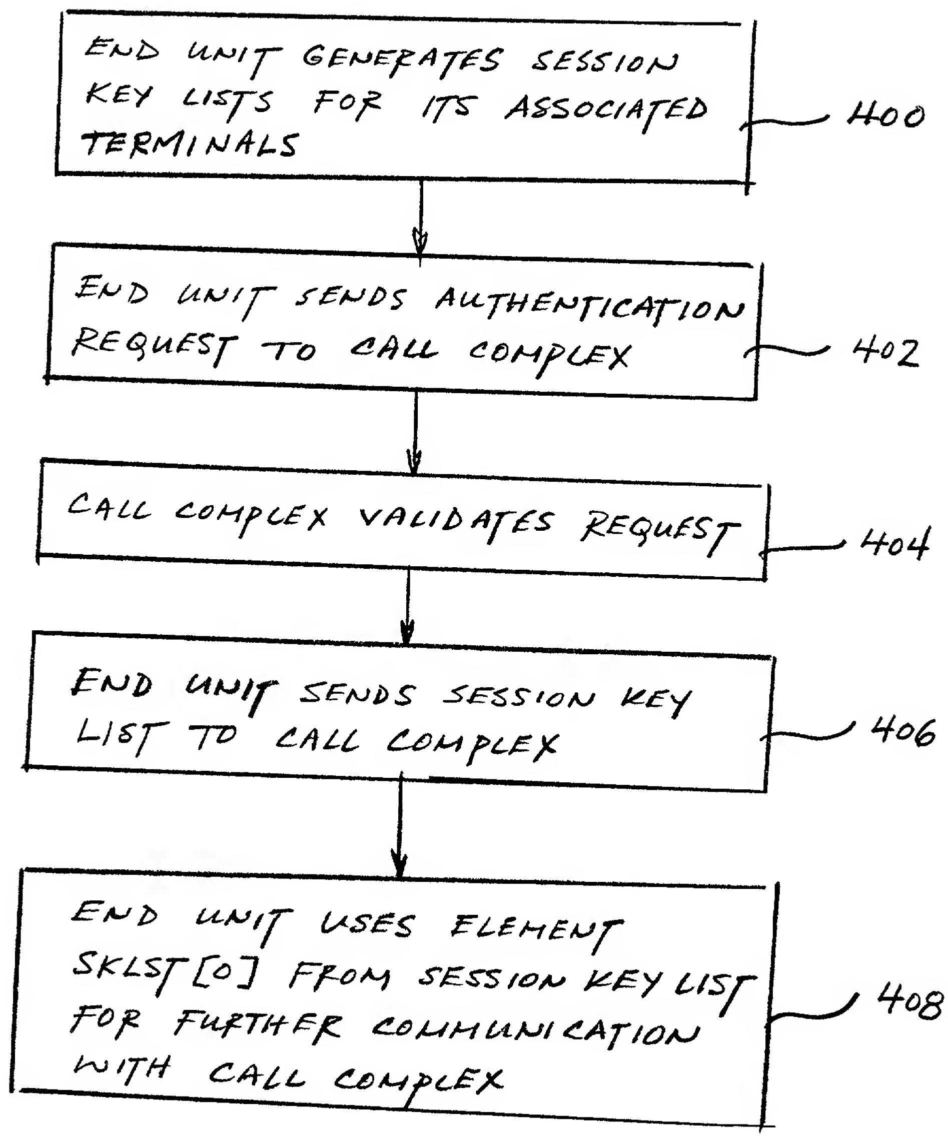
END UNIT GENERATES SESSION KEY LISTS FOR ITS ASSOCIATED TERMINALS
400
END UNIT SENDS AUTHENTICATION REQUEST TO CALL COMPLEX
402
CALL COMPLEX VALIDATES REQUEST
404
END UNIT SENDS SESSION KEY LIST TO CALL COMPLEX
406
END UNIT USES ELEMENT SKLST[0] FROM SESSION KEY LIST FOR FURTHER COMMUNICATION WITH CALL COMPLEX
408

FIG. 4

Call Complex		End Unit
	←	End Unit $_{\text{Session Key}}$ = Random() ESKe = Encrypt (End Unit $_{\text{Session Key}}$) End Unit $_{\text{Private Key}}$ EEUIDe = Encrypt (End Unit Identification) Call Complex $_{\text{Public Key}}$ SendAuthenticationReq(EEUIDe,ESKe)
Identify Request (Validate request; if it is not valid, drop it) End Unit Identification = Decrypt(EEUIDe) Call Complex $_{\text{Private Key}}$ If (End Unit Identification) exists get End Unit $_{\text{Public Key}}$ End Unit $_{\text{Session Key}}$ = Encrypt (End Unit $_{\text{Session Key}}$) End Unit $_{\text{Public Key}}$ ACKe = Encrypt(ACK) End Unit $_{\text{Session Key}}$ CreateSessionInformation(IP Address, End Unit Identification) SendRegistrationAcknowledgment(ACKe)	→	
		SKLSTe = Encrypt(GenerateSessionKeyListForEndUnit())End Unit $_{\text{Session Key}}$ SendSessionKeyList(SKLSTe)
SKLST = Decrypt(SKLSTe) End Unit $_{\text{Session Key}}$ =SKLST[0] ACKe = Encrypt(ACK) End Unit $_{\text{Session Key}}$ SendSessionKeyListAcknowledgment(ACKe)		End Unit $_{\text{Session Key}}$ =SKLST[0]

FIG. 5

Call Complex		End Unit 1
	←	CallRequestTo(Extension 201, Extension 105) End Unit $_{\text{Session Key}}$
If incoming Request IP Address not registered, drop the request End Unit $_{\text{Session Key}}$ = Find Session Key for IP(Request IP Address) Call Request Data = Decrypt (Incoming Buffer) End Unit $_{\text{Session Key}}$ If Plaintext Buffer does not contain End Unit Registration name, drop the request		
Call Complex		**End Unit 2**
EUEUSK = SKLST[105] Message Key = get_key_for_extension(201) SendIncommingCallRequest(Encrypt (oIP,201,105,EUEUSK) $_{\text{Message Key}}$)	→	
	←	If Incoming Request IP Address not Call Complex, drop the request Plaintext Buffer = Decrypt(Incoming Buffer) End Unit $_{\text{Session Key}}$ If Plaintext Buffer does not contain End Unit Registration name, drop the request Set EUEUSK SendCallAcceptedInformation(RTP info) Unit $_{\text{Session Key}}$
End Unit 1		**End Unit 2**
SendVoicePacket(Encrypt(Plaintext Buffer) $_{\text{EUEUSK}}$) ReceiveVoicePacket(Decrypt(Incoming Buffer)) $_{\text{EUEUSK}}$)	↔	ReceiveVoicePacket(Decrypt(Incoming Buffer)) $_{\text{EUEUSK}}$) SendVoicePacket(Encrypt(Plaintext Buffer) $_{\text{EUEUSK}}$)

FIG. 6A

End Unit 1		Call Complex
ConfRequestTo(Extension 311, Extension 105) End Unit $_{\text{Session Key}}$	→	If incoming Request IP Address not registered, drop the request End Unit $_{\text{Session Key}}$ = Find Session Key for IP(Request IP Address) Call Request Data = Decrypt (Incoming Buffer) End Unit $_{\text{Session Key}}$ If Incoming Buffer does not contain End Unit Registration name, drop the request
End Unit 3		**Call Complex**
	←	EUEUSK = SKLST[105] Message Key = get_key_for_extension(311) SendIncommingConfRequest(Encrypt (oIP,511,105,EUEUSK) $_{\text{Message Key}}$)
If Incoming Request IP Address not Call Complex, drop the request Buffer = Decrypt(Incoming Buffer) End Unit $_{\text{Session Key}}$ If Plaintext Buffer does not contain End Unit Registration name, drop the request Set EUEUSK SendConfAcceptedInformation(RTP info) Unit $_{\text{Session Key}}$		
End Unit 3		**End Unit 1**
SendVoicePacket(Encrypt(Plaintext Buffer) $_{\text{EUEUSK}}$) ReceiveVoicePacket(Decrypt(Incoming Buffer)) $_{\text{EUEUSK}}$)	↔	ReceiveVoicePacket(Decrypt(Incoming Buffer)) $_{\text{EUEUSK}}$) SendVoicePacket(Encrypt(Plaintext Buffer) $_{\text{EUEUSK}}$)
End Unit 3		**End Unit 2**
SendVoicePacket(Encrypt(Plaintext Buffer) $_{\text{EUEUSK}}$) ReceiveVoicePacket(Decrypt(Incoming Buffer)) $_{\text{EUEUSK}}$)	↔	ReceiveVoicePacket(Decrypt(Incoming Buffer)) $_{\text{EUEUSK}}$) SendVoicePacket(Encrypt(Plaintext Buffer) $_{\text{EUEUSK}}$)

FIG. 6B

End Unit 1		Call Complex
DropSession(Extension 311) End Unit $_{Session\ Key}$	→	If Incoming Request IP Address not registered, drop the request End Unit $_{Session\ Key}$ = Find Session Key for IP(Request IP Address) Call Request Data = Decrypt (Incoming Buffer) End Unit $_{Session\ Key}$ If Plaintext Buffer does not contain End Unit Registration name, drop the request
End Unit 3		**Call Complex**
CleanUp()	←	DropSession(Extension 311) End Unit $_{Session\ Key}$

FIG. 6C

End Unit 2		Call Complex
	←	EUEUSK-NEW = SKLST[105,NEXT] // GET NEXT SESSION KEY FROM EXTENTION 105 STACK Message Key = get_key_for_extension(201) SendNewSessionKeyRequest(Encrypt (oIP,201,105, EUEUSK) $_{\text{Message Key}}$)
If incoming Request IP Address not Call Complex, drop the request Plaintext Buffer = Decrypt(Incoming Buffer) End Unit $_{\text{Session Key}}$ If Plaintext Buffer does not contain End Unit Registration name, drop the request Set EUEUSK to EUEUSK-NEW SendConfForNewSessionKeyRequest() Unit $_{\text{Session Key}}$		
End Unit 1		**End Unit 2**
SendVoicePacket(Encrypt(Plaintext Buffer) $_{\text{EUEUSK-NEW}}$) ReceiveVoicePacket(Decrypt(Incoming Buffer)) $_{\text{EUEUSK-NEW}}$)	↔	ReceiveVoicePacket(Decrypt(Incoming Buffer)) $_{\text{EUEUSK -NEW}}$) SendVoicePacket(Encrypt(Plaintext Buffer) $_{\text{EUEUSK-NEW}}$)
End Unit 1		**End Unit 2**
EndOfSession(Encrypt(Plaintext Buffer) $_{\text{EUEUSK-NEW}}$)	→	CleanUp()
Call Complex		**End Unit 1**

	←	End Unit 105 $_{\text{Session Key}}$ = Random() // Create a new session key for 105
		EUSKe = Encrypt (EUSN, End Unit 105 $_{\text{Session Key}}$) End Unit $_{\text{Private Key}}$
		SendSessionKey(EUSKe)

If incoming Request IP Address not registered, drop the request End Unit $_{\text{Session Key}}$ = Find Session Key for IP(Request IP Address) Call Request Data = Decrypt (Incoming Buffer) End Unit $_{\text{Session Key}}$ If Plaintext Buffer does not contain End Unit Registration name, drop the request Update SKLST[105] = End Unit 105 $_{\text{Session Key}}$ // This is a stack operation; new key is first available key in the stack		

	←	End Unit 105 $_{\text{Session Key}}$ = Random() // Create a second new session key for 105
		EUSKe = Encrypt (EUSN, End Unit 105 $_{\text{Session Key}}$) End Unit $_{\text{Private Key}}$
		SendSessionKey(EUSKe)

If incoming Request IP Address not registered, drop the request End Unit $_{\text{Session Key}}$ = Find Session Key for IP(Request IP Address) Call Request Data = Decrypt (Incoming Buffer) End Unit $_{\text{Session Key}}$ If Plaintext Buffer does not contain End Unit Registration name, drop the request Update SKLST[105] = End Unit 105 $_{\text{Session Key}}$ // This is a stack operation; new key is first availible key in the stack		

FIG. 6D